Microservice Architecture: Optimizing for speed

An in-depth article upon what Microservice Architecture stands for and with it what pros and cons it brings along.

**“Microservice Architecture is a term used to define the procedure of dividing up an application into a series of smaller and more specified parts, where each part communicates with another through common interfaces.”**

To know more about microservices we must know a bit of the background story.

# Background

In the early stages of computer revolution, using almost any computer required writing a custom software. Only a Ph-D in science and computer could use these programing languages and entry into these programing languages was a tremendous task.

In the 1960s, the use of computer application skyrocketed and thus in 1964, **Basic** (a general-purpose programming language) was developed lowering the barriers for the entry in the programming allowing students without Ph-D to write executable programs.

The growth also brought forth a complexity of Software System which was overcome by the old times method of **Divide and Conquer**

1970s brought Modular Software development through the works of people such as **Edsger W. Dijikstra** (who in 1972 introduced the concept of Separation of Concern) and David **Parnas** (for his idea of modularity and information hiding in software’s for his paper of 1972).

This gave the idea of decomposing a large, complex software system into **“Loosely coupled, highly cohesive”** modules which communicated via internal interfaces.

Loosely coupled mean the dependency between modules should be very low.

Highly cohesive means that the that mono module should focus on single or similar functionality.

The rise of the internet and web in the 1990s software became widespread in business applications and became even more complex and large. Although modularity is used to reduce the complexities of the software applications, but often it did not help as the soft modular boundaries of software sub-system are easy to cross and misuse. **Layered Architecture** was another software architecture pattern that became very popular in the 1990s to develop business applications.

An ideal business Web Applications is divided into several layers as shown below:

A screenshot of a cell phone

Description automatically generated Layered Architecture

As a result of Mobile internet and faster networks the late 2000s saw Cambrian Explosion. The software took over the world with a storm and all types of business started going digital such as Banking, Hotels, Music etc., and companies such as Facebook, Twitter, Uber, Netflix, Spotify came with such innovative, aggressive approach that the Monolithic Architecture could not handle the challenges being thrown at them. It was tine for a more efficient approach.

To know what need to improve we must know what the limitation were being faced.

# Limitations of Monolithic Architecture

## Application Scaling

As the successful Web Scale companies see exponential growth their software isn’t able to keep up with their need for support high horizontal scalability and where monolithic software works as a single unit and developed in a single programming language using a single Tech Stack an architecture was needed that could support the polyglot programming .

A close up of a device

Description automatically generated

If horizontal scaling is desired then the whole application needs to be scaled and with monolithic software only supporting one programming language, we are not able to implement one single module of it in other programming languages.